# Creating Flexible Mappings between Building Information Models and Cost Information

Michael Lawrence [a], Rachel Pottinger [b], Sheryl Staub-French [c,] *, and Madhav Prasad Nepal [d]

[a] Google Inc., 747 6t St S, Kirkland, WA 98033, United States; Email: mlawrence@google.com

[b] Department of Computer Science, University of British Columbia, Vancouver, BC, Canada V6T 1Z4; Email: rap@cs.ubc.ca

[c] Department of Civil Engineering, University of British Columbia, Vancouver, BC, Canada V6T 1Z4; Email: sherylsf@civil.ubc.ca; Tel: +1 604 827 5118

[d] School of Civil Engineering and Built Environment, Queensland University of Technology, 2 George St GPO Box 2434, Brisbane, QLD 4001, Australia; Email: madhav.nepal@qut.edu.au

*** Corresponding author**

## ABSTRACT

During the early design stages of construction projects, accurate and timely cost feedback is critical to design decision making. This is particularly challenging for cost estimators, as they must quickly and accurately estimate the cost of the building when the design is still incomplete and evolving. State-of-the-art software tools typically use a rule-based approach to generate detailed quantities from the design details present in a building model and relate them to the cost items in a cost estimating database. In this paper, we propose a generic approach for creating and maintaining a cost estimate using flexible mappings between a building model and a cost estimate. The approach uses queries on the building design that are used to populate views, and each view is then associated with one or more cost items. The benefit of this approach is that the flexibility of modern query languages allows the estimator to encode a broad variety of relationships between the design and estimate. It also avoids the use of a common standard to which both the designers and estimators must conform, allowing the estimator added flexibility and functionality to their work.



**Keywords:** Mapping, BIM, Query, Cost estimate, Design changes, IFC, Views, XML, Schema, Database


## 1. Introduction

Accurate and timely cost feedback is critical for design decision-making on building construction projects. In the early phases of design, this is a significant challenge for estimators as they must create detailed and accurate cost estimates although the design is still evolving and changing and there is often incomplete information. Repeated changes in design parameters, component types and materials often occur to meet a project's budget, performance requirements, and to improve constructability. These design changes not only necessitate changes in material quantities, they may also impact other aspects of the cost estimate that are more subtle and difficult to detect, such as changes to labour productivity rates, construction methods, and related cost items. Currently, determining how a design has changed and how it impacts construction costs is a laborious and largely manual task for cost estimators.

Building Information Modeling (BIM) is an increasingly popular enabling technology for digitally representing building information and supporting information exchange in the architecture, engineering, construction and facilities management (AEC/FM) industry. BIM standards are strongly product centric and oriented towards design practitioners. Extracting relevant information for construction practitioners, and in particular, in support of cost estimating remains a challenge. It is important to emphasize that one challenge is that BIM and cost estimating approaches use different *schemas* – representations of the data. For example, the BIM schema in Fig. 10 is very different to the cost estimate in Fig. 11. This is inherent in the problem

– because the BIM and cost estimation communities have different sets of vocabularies, there is no one overriding schema that describes all, as we discuss in more detail when we describe related research (Section 3).

Existing approaches and software applications for BIM-based cost estimating attempt to automate the process of quantity takeoff (creating and managing sets of material quantities) and the selection of labour items and equipment needed to construct building components of a specific type. These approaches however suffer from the following limitations:

- Their dependence on detailed component information limits their usability during the early phases of design. For example, millwork specifications may not be given until much later in the design phase.
- Their rule-based approach is rigid and does not consider the cost-estimating impact of more abstract or subtle construction conditions. For example, the number of concrete columns offset from major grid lines may be relevant to the unit rate of column formwork.
- They are strongly coupled to the choice of design software, meaning the estimator's process must be tailored to suit the designer's. As discussed in [1], there is considerable variation in the procedures used to create a BIM by different design firms.
- They do not handle design changes beyond changes to material quantities well. Cost estimators are often interested in knowing the specific differences between successive design revisions and their ramification on cost estimates.
- The process of estimating involves more than just extracting counts and measurements. It involves assessing conditions, such as unusual wall conditions, unique assemblies, and difficult access conditions in the project that impact cost. Automatic identification of these conditions by any BIM tool remains a big challenge [2].

We address these limitations by proposing a generic approach for creating and maintaining a cost estimate by using mappings to relate cost information to a BIM. Our approach executes queries (i.e., formal instructions that tell the computer how to answer questions), both over the building design, and views (i.e., saved query results) over the design data. This approach of using views is an established mechanism for allowing independence between the models/standards used on both the design and estimate [3, 4]. We propose a system whereby the cost estimator creates queries on the building design that are used to populate views, and each view is associated with one or more cost items. While creating a query of this sort may be difficult for an estimator, there are tools that can help with this process [5], and our future work includes having a graphical user interface where these queries are hidden below the surface. However, the benefit of this approach is huge: the flexibility of modern query languages allows the estimator to encode a broad variety of relationships between the design and estimate – beyond the typical case of material quantities. It also eschews the use of a common standard to which both the

designers and estimators must conform, allowing the estimator to add on functionality to their existing process.

The remainder of this paper is organized as follows. In Section 2 we introduce the study which motivated our research, and describe the application of modern BIM based cost estimating software to this case. In Section 3, we review recent literature on cost estimating in BIM, and outline the tenets guiding our approach. Section 4 describes the details of our approach and the types of mappings it can handle. The aspect of system design and operation is discussed in Section 5. In Section 6, we describe a user study that we designed based on the case study introduced in Section 2 to demonstrate the benefits of our system and gather additional insights, and conclusions are then provided in Section 7.

## 2. The case study

Our approach is guided by a case study of a recently constructed building in British Columbia, Canada. This facility is a 93,000 SF, $31M building which includes a lecture theatre, office space, wet and dry laboratories, and animal care facilities. Fig. 1 shows a 3D rendering of the case study project. We obtained two detailed cost estimates (CP1 and CP2), prepared at 16 weeks apart in response to the changes in design, as well as the architectural and structural drawings corresponding to CP2. The cost estimates are tables of cost items organized according to the MasterFormat standard and each item has the following attributes: code, description, quantity, unit of measure, rate per unit, and total cost. We also obtained the files used to generate material takeoffs using *On-Screen Takeoff*, which is a program for calculating quantities based on digital drawing sheets (this is discussed in more detail below). The architectural and structural drawings are both electronically annotated using *On-Screen Takeoff*. We analyzed the differences between the two cost estimates, and interviewed the cost estimators familiar with the project working for the general contractor. We confirmed how the design had changed between CP1 and CP2, as well as details on the quantity takeoff and cost estimating process used for this project. We recreated a subset of the building design in a BIM using Autodesk *Revit*, incorporating all of the details (e.g. material layers) specified in the drawings into our model, which was then exported as Industry Foundation Classes (IFC) and converted into ifcXML. We used the case study BIM to evaluate state-of-the-art BIM-based applications and the ifcXML version of the case study to test our prototype application.

**Fig. 1.** 3D Rendering of the case study project.

The cost estimator for the case study relied on *On-Screen Takeoff* to calculate material quantities from a digital set of building drawings. The estimator created a number of "conditions", each corresponding to a type of component in the design. The estimator then annotated the drawings (lines, polygons or points) to represent quantities of interest (areas, lengths, etc.) on the drawings and associated each with a condition. Fig. 2 shows a snapshot of a portion of the structural plan for the basement level, where a number of conditions have been created for slab on grade concrete, footings of various types and walls.

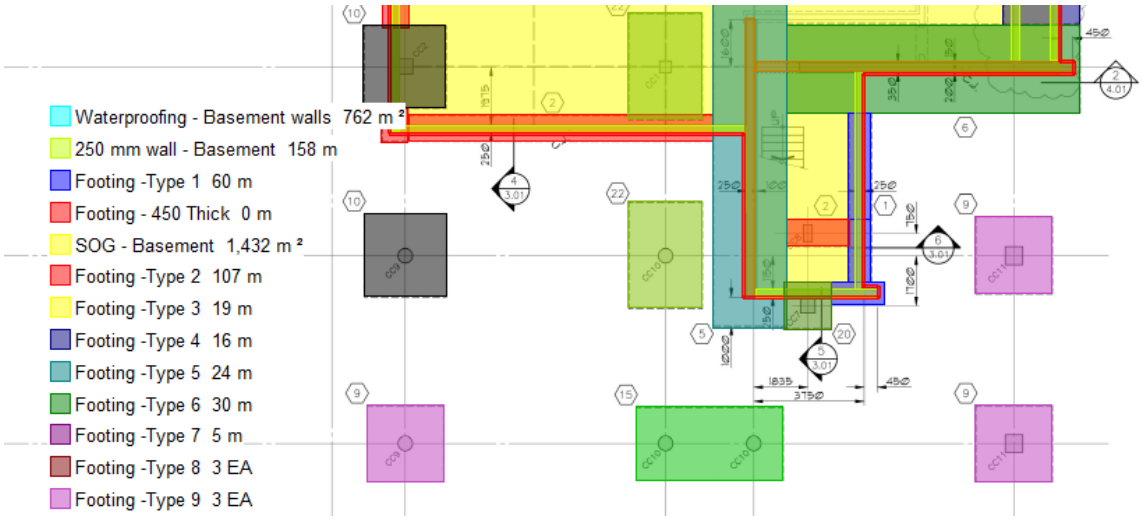


**Fig. 2.** Screen shot of *On-Screen Takeoff* conditions used to takeoff footing concrete from a structural drawing.

Creating these conditions allows for a detailed takeoff to be performed, as the necessary quantities are calculated automatically from the visual conditions. Fig. 3 shows the quantities derived for a few of the footing items. In most cases, their perimeter, area, and volume are given.

| No. | Name | Height | Area | Quantity 1 | UOM1 | Quantity 2 | UOM2 | Quantity 3 | UOM3 |
|---|---|---|---|---|---|---|---|---|---|
| Concrete Footings | | | | | | | | | |
| 122 | Footing -Type 1 | 300 mm | **Multi-Area Total** | **79** | **m** | **47** | **m²** | **18** | **m³** |
| 135 | Footing - 450 Thick | 0 mm | Basement | 0 | | 16 | m | 6 | m³ |
| 227 | Footing -Type 2 | 300 mm | **Multi-Area Total** | **168** | **m** | **101** | **m²** | **45** | **m³** |
| 228 | Footing -Type 3 | 350 mm | **Multi-Area Total** | **65** | **m** | **45** | **m²** | **27** | **m³** |
| 229 | Footing -Type 4 | 350 mm | **Multi-Area Total** | **32** | **m** | **23** | **m²** | **29** | **m³** |
| 230 | Footing -Type 5 | 750 mm | Basement | 24 | m | 36 | m² | 45 | m³ |
| 231 | Footing -Type 6 | 750 mm | **Multi-Area Total** | **36** | **m** | **54** | **m²** | **81** | **m³** |
| 232 | Footing -Type 7 | 850 mm | **Multi-Area Total** | **12** | **m** | **21** | **m²** | **39** | **m³** |
| 233 | Footing -Type 8 | 500 mm | **Multi-Area Total** | **5** | **EA** | **24** | **m²** | **14** | **m³** |
| 234 | Footing -Type 9 | 600 mm | **Multi-Area Total** | **5** | **EA** | **31** | **m²** | **20** | **m³** |
| 235 | Footing -Type 10 | 600 mm | Basement | 15 | EA | 101 | m² | 71 | m³ |

**Fig. 3.** Quantities calculated automatically from the visual conditions shown in Fig. 2.

For simplicity, we will focus on a few items related to concrete footings, which includes items for footing forms, rebar, and concrete. In order to specify quantities for these items, the cost estimator must add up a measure of all footings (although we only show a few, there are 20 such items). In the case of footing forms, the cost estimator must manually calculate the sum of all footing areas, and convert it to imperial units. In the case of footing concrete, the same must be done for all footing volumes. In the case of rebar, a formula is applied to the volume (the conversion used is 200lb per cubic yard). This process of aggregating these different quantities for each cost item is time-consuming and laborious. Moreover, except for creating the conditions, these steps must be repeated when an updated design is given.

In interviews with cost estimators, it was noted that one of the major challenges with their current process is updating the estimate to correspond to changes in the design. In this project and other projects, the building design underwent multiple revisions before construction was started, and each revision was normally issued without any clear and precise indication of what has changed. Each revision, therefore, results in a lot of manual work reviewing the new drawings looking for changes, and updating the *On-Screen Takeoff* quantities as appropriate.

### *2.1 BIM-based cost estimating*

BIM-based estimating applications address many of the weaknesses with current techniques that rely on 2D paper or digital drawings. We evaluated a state-of-the-art BIM-based estimating application called *Innovaya* [6], which imports design information from Autodesk *Revit*, *AutoCAD*, or *Tekla*. When a design is imported, a set of "managed quantities" can easily and automatically be created. Each managed quantity corresponds to a type of component in the design. These quantities can be exported to an Excel spreadsheet or combined with a cost database from *MC²-ICE* or *Timberline* to create an estimate. Fig. 4 shows the result of associating a type of interior partition wall of another recently constructed building modeled in *Revit* to a *Timberline* assembly. In this example, the height and total length of walls of type *Interior – 4 7/8" Partition (1-hr)* are mapped to the length and height of the *Timberline* assembly

*0932 – Wall – S Studs Interior*, as indicated by the "Mapping" column in the top-right section. *Innovaya* remembers this mapping, so that when the design is changed, the quantities of the assemblies and items are automatically updated to reflect this.

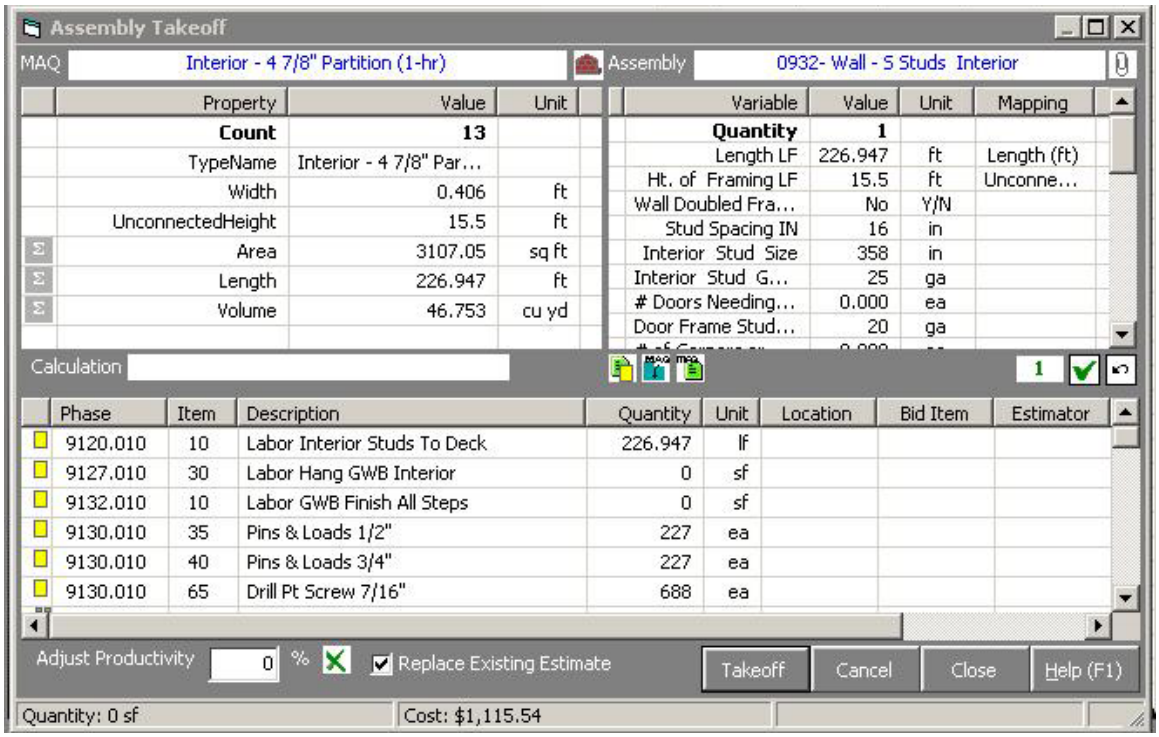


**Fig. 4.** Using *Innovaya* to map a managed quantity.

While *Innovaya* provides support to automatically generate quantities out of BIM, we found *Innovaya*'s method of grouping building components into managed quantities less useful for estimating some assembly types. We explored the use of *Innovaya* on another recently constructed building, which contains a large glazed curtain wall, and found a large number of groups corresponding to identical types of components. Many other types of architectural objects that are logically a single unit ended up being represented by a large number of managed quantities in *Innovaya*. This created difficulty in matching managed quantities to assemblies. Based on our experience, we concluded that the usefulness of *Innovaya* for BIM based takeoff depends on how types are organized in the BIM. Specifically, the grouping of building components into types in the BIM needs to match the grouping of quantities into cost items used by the cost estimator. If this is not the case, the estimator will need to manually edit the quantities. As pointed out in [1], the lack of industry wide standards for object definitions in BIM reduces its usefulness for the cost estimator. While standard schemas such as IFC specify a common format that can be read by many applications, the semantics may vary considerably depending on who produced the model and with what software. For example, one architect might use wall types W1, W2, and W3, while another uses 2099, 2089, and 2091. While *Innovaya* and other BIM-based applications have made significant progress in better mapping the relationships

between design and cost information, there are other types of important mappings that have not been addressed by state-of-the-art software, which are described in the next section.

### *2.2 Mapping types*

Here we characterize the types of mappings which are often overlooked in the existing BIM-based applications but are extremely important for coordinating and updating cost estimates, especially when there are changes in design.

#### *2.2.1 Proxies*

One of the mapping features which we found would be useful for cost estimators is to use "proxies" to relate a design to the cost estimate. During the early phase of a building project, exact details are often not available, and lump sum estimates are often created. For example, the estimates CP1 and CP2 were composed during the building's early design phase, where some details are unknown, yet must be accounted for in the estimate. Consider the items relating to millwork from CP2 as shown in Fig. 5.

| | | | | | | |
|---|---|---|---|---|---|---|
| **6220 - Millwork** | | | | | **$** | **323,137** |
| 0010 | Millwork - Lecture Theatre (Lectum Desk) | 1 | ea | 13,000.00 | $ | 13,000 |
| 0010 | Millwork - Projection Room | 1 | Lsum | 4,000.00 | $ | 4,000 |
| 0010 | Millwork - Classroom (Lectum desk) | 1 | ea | 13,000.00 | $ | 13,000 |
| 0010 | Millwork - Custom millwork in Board room (RM 470) | 1 | Lsum | 20,000.00 | $ | 20,000 |

**Fig. 5.** Items relating to millwork in the UBCO cost estimate.

The cost estimator created rough "lump sum" estimates for millwork in a few of the building's rooms. Exact millwork details were not available at this time, but the estimator may use some related building features which are available as *proxies*. In this example, the cost of lecture theatre millwork is related to the gross area of the lecture theatre; if the lecture theatre changes in size, the cost of this item is likely to need updating as well.

#### *2.2.2 Aggregated conditions*

Aggregated conditions characterize clusters of building components or design features beyond the itemized BIM objects or proxies. These conditions can be expressed as an aggregation over all building components and mapped to an estimate. For example, if more than 50% of interior partition walls exceed a height of 10 feet, then the rate per square foot of partition walls in the estimate may need to be increased because additional equipment is required (e.g. ladders) and the productivity of the installation crew is reduced.

#### *2.2.3 Spatial conditions*

Certain spatial and geometric configurations of building components can have an dramatic effect on the cost and constructability of a design. For example, as shown in [7], the overall uniformity of column sizes from floor to floor effects the difficulty of constructing formwork for

cast in place concrete columns. The greater the variation in column sizes, the work is less repetitive and productivity is reduced. If the distribution of column sizes changes between successive design revisions, the estimator may want to update the cost for related items.

Our approach, described in Section 4, is able to represent these more complex relationships between design and cost information using spatial and non-spatial queries over BIM data to express an estimator's rationale on how and when a given design impacts construction costs. These queries can be related to specific cost items in an estimating database, so that the cost estimator can pinpoint how design changes may affect the estimate even when the design is still evolving. Our approach is general and yet flexible in that it is independent of the BIM software and the procedures used to develop a BIM.

## 3. Related research

There is significant potential in utilizing BIM for cost estimating. For example, in [8], significant cost savings was achieved through changing processes for cost estimating in building information modeling. Shen and Issa [9] performed a user study demonstrating the benefit of aggregating material quantities from IFC for cost estimating. Despite these examples, numerous studies have indicated that cost estimators have not adopted BIM as enthusiastically as other participants in an AEC project. McCuen [10] points to a number of contributing factors in the slow uptake of BIM by cost estimators, most notably the lack of software interoperability and standards. Hannon [11] points out that the skill required to map objects in a BIM to take-off quantities is not possessed by cost estimators accustomed to traditional practices.

Attempts to marry BIM with the cost estimation process tend to focus on defining standards for supporting the task of quantity takeoff, which as we argued in Section 2 is only part of the picture. In [12], the author describes efforts aimed at defining what BIM must contain, and how exchange takes place with respect to quantity takeoff. These attempts tend to approach the problem by defining and expanding BIM standards to include sufficient detail for supporting the task of cost estimating, for example by including cost information in the design.

Although we believe standards are an important mechanism to the exchange of information and we support the idea of expanding BIM to include design details sufficient for cost estimating, our approach is guided by a skeptical attitude towards the success of large standards in supporting complicated tasks for a broad variety of participants. As argued in [13], [14], IFC's failure is that it attempts to be overly general, resulting in substantial redundancy and ambiguity in how it is implemented. Kraus et al. point out that this ambiguity permits inconsistency with how BIM is used across different design firms [1]. Hartmann et al. argue that the loosely coupled structure of the construction industry limits the success of top-down technology pushing, and that a "pull" approach that aligns existing BIM tools with current work practices is more likely to be successful [15].

Given this position, it is important to emphasize that our work does *not* create another schema to compete with existing schemas. Rather, our work allows queries over existing schemas to improve interoperation between them. While the examples in this paper are limited to a small number of schemas (e.g., ifcXML), the approach outlined in this paper is by no means limited to these schemas. Indeed, it could be used in entirely different scenarios, such as coordinating economic indicators between a national department of finance and local statistic gathering bodies [16, 17].

Some current approaches work on the data level – each individual item in the design would be mapped to an individual item in the estimate. Or work is on the schema level – the representation of how the data is stored. This allows for a huge speed up in efficiency [18]. For example, instead of requiring mapping between each individual wall and its cost, we can describe the cost of all walls of that type and the cost per wall. This also allows designers to be more flexible in their design changes, since changes can be reasoned about at a higher level.

In the database research community, interoperability has been a long standing problem. A typical approach to integrating heterogeneous databases is to define a common schema and mappings between the individual databases and this common schema (see [19] for a survey). This approach requires substantial up-front effort before any benefit can be gained from the integration. Recently, Halevy et al. have proposed the idea of a "pay as you go" data integration system, whereby mappings can be added incrementally as they are needed [20]. Our approach to integrating building design and cost information follows this pay as you go philosophy. It is our assertion that the complexity of cost estimating is a significant barrier to defining a standard which is suitable for most practitioners. We prefer allowing these individual users to define the relationships and semantics that are important to them, and to provide as much automated support in this task as possible. Not only does our approach respect the existing cost estimating processes in place, it also allows benefit to be derived from early phase models with missing data, such as our theatre millwork example from Section 2.

Interoperability has been studied in other communities as well. For example, the HUMBOLDT data harmonization framework [21] was a project funded by the European Union to harmonize geographic data. Like our work, HUMBOLDT's mapping creator, Hale [21] maps between data sources at the schema level rather than at the instance level. However, like many of the approaches in the database research community, their work assumes that the data is represented in a set of transformations that is fairly limited; our mappings are more complex – they associate a query over one schema with a query over another schema, enabling us to be more expressive.

## 4. A generic, query-driven approach

We propose a system which operates independently from whatever software systems are being used by the designer and cost estimator. The central idea in our system is that the cost

estimator will create mappings to express how their cost estimate relates to the building design (BIM). A mapping associates a query on the building design with some attribute(s) of some item(s) in the cost estimate; a query is a statement that expresses a function computing some set of values given a BIM as input.

A view refers to a saved query or its output. Views are an important mechanism for, among other things, hiding the complexity of an underlying database from applications that access it [22]. Views are also commonly used for integrating heterogeneous databases, and the types of mappings we use were first proposed by Friedman et al. [23]. Recently, the use of views (sometimes referred to as model views) has been proposed to address the problem of application interoperability by adding a layer of specificity on top of the ambiguity and redundancy of IFC [14], [13].

Returning to the theatre millwork example from Section 2.2, recall that our cost estimator uses theatre area as a proxy for the cost of theatre millwork. Fig. 6 shows a snippet of the XML (a simplified version of ifcXML) corresponding to the BIM we have created for the building described in Section 2. Using this BIM, our estimator would create a mapping by formulating a query (in the XQuery language) on this building design which extracts the area of the Theatre, as:

//SPACE[LONGNAME='Theatre']/PROPERTYSET/PROPERTY[NAME='Area']/VALUE.

This query would return the value 7442.34. The mapping would associate this query and its corresponding return value with the "Millwork – Lecture Theatre (Lecturn Desk)" cost item in the cost estimate as shown in Fig. 5. When the building design XML is changed, our system would reevaluate this query, and if the value changes (e.g. to 8400), then the estimator will be notified that "Millwork – Lecture Theatre (Lecturn Desk)" may need updating, along with any other cost items that have been associated with this query.

A design can be represented in many possible BIM formats. It should be mentioned that our approach does not depend on the specific BIM format, although we mention IFC and ifcXML in the process of creating mappings. Our example used the XQuery language on a simplified version of ifcXML to represent the BIM data; other possibilities are the EXPRESS-V language for views in the EXPRESS data model proposed by Spooner and Hardwick [3], and the Structured Query Language (SQL) used for relational databases (such as Oracle Spatial), extended with BIM-specific operators as in [24]. While we did not use EXPRESS or SQL in this example, our system is easy to extend to these (and) other query languages. Indeed, the example shown in Fig. 8 uses SQL as this was the best choice of query language to use for the sample data that we had available.

Another feature of our system is the types of queries it can support. In Section 2.2, we discussed the types of mappings that are important for cost estimating. Consequently, we broadly categorize the types of queries we are able to support into the following:

- Aggregated Quantities
- Proxies
- Spatial Queries

```
<COLUMN>
  <NAME>M_Rectangular Column:CC3:124568</NAME>
  <OBJECTTYPE>CC3</OBJECTTYPE>
  <TAG>124568</TAG>
  <PROPERTYSET>
    <NAME>PSet_Revit_Dimensions</NAME>
    <PROPERTY>
      <NAME>Depth</NAME>
      <VALUE>1.31</VALUE>
    </PROPERTY>
    <PROPERTY>
      <NAME>Width</NAME>
      <VALUE>1.31</VALUE>
    </PROPERTY>
  </PROPERTYSET>
</COLUMN>
<COLUMN>…</COLUMN>
<SPACE>
  <NAME>196</NAME>
  <LONGNAME>Theatre</LONGNAME>
  <INTERIOROREXTERIORSPACE>internal</INTERIOROREXTERIORSPACE>
  <PROPERTYSET>
    <NAME>PSet_Revit_Dimensions</NAME>
    <PROPERTY>
      <NAME>Area</NAME>
      <VALUE>7442.34</VALUE>
    </PROPERTY>
    <PROPERTY>
      <NAME>Perimeter</NAME>
      <VALUE>358.69</VALUE>
    </PROPERTY>
    <PROPERTY>
      <NAME>Unbounded Height</NAME>
      <VALUE>22.12</VALUE>
    </PROPERTY>
  </PROPERTYSET>
</SPACE>
<SPACE>…</SPACE>
```

**Fig. 6.** An example building design in XML consisting of a few columns and spaces.

*4.1 Aggregated quantities*

Extracting material quantities is the common feature of current state-of-the-art BIM-based estimating applications in use. Using the BIM created for the building we studied in Section 2, we were able to create a number of queries extracting various quantities. For example, the following XQuery statement extracts the total area of all suspended slabs:

SUM(

```
FOR $X IN //SLAB
WHERE $X/PROPERTYSET/PROPERTY/VALUE = ‘Suspended Slab’
RETURN DATA ($X/PROPERTYSET/PROPERTY[NAME='Area']/VALUE.
```

The estimator would create a mapping by associating this query with the quantity of the “Suspended Slab” cost item. When the BIM is updated, our system would automatically update this cost item, if necessary.

*4.2 Proxies*

We have already mentioned the importance of proxies during the early design phase, when an estimate is needed but exact details are missing from the design. Using our system, the estimator can formulate queries for proxies, which represent values extracted from the design that are used in lieu of the specific details, as in the theatre area and theatre millwork example used throughout this paper. In our example from Section 2, there are other cost items which depend on details which are not yet available. For example, the cost of metal stairs has been estimated as a lump sum item. If the floor-to-floor height of the building should change, or the number of stairwells in the design, then the cost of metal stairs may be affected. The estimator can formulate queries for these attributes of the design and relate them to the metal stairs cost item in a mapping.

*4.3 Spatial queries*

Certain overall properties of the building design can impact the construction process and the resulting cost in a way that is independent from material quantities. Spatial queries refer to a broad class of queries which identify these overall properties or conditions. For example, in [7], we proposed the use of spatial BIM queries to examine constructability conditions such as the number and location of wall penetrations, and the minimum column spacing. Because these features are implicitly represented in the design, they are not accounted for by traditional BIM-based quantity takeoff approaches. Our approach is able to use these queries as part of the estimating process.

## 5. System design and operation

An IDEF0 diagram illustrating the overall operation of our system is given in Fig. 7. In our prototype system, the input BIM data is represented using the ifcXML data standard. The cost estimate data is represented using tables of cost items that are organized according to the MasterFormat standard consistent with the case study example presented in Section 2. Note that activities A0 through A2 are performed for each new building that needs to be estimated, and activities C0 and C1 are performed each time a building design is updated.

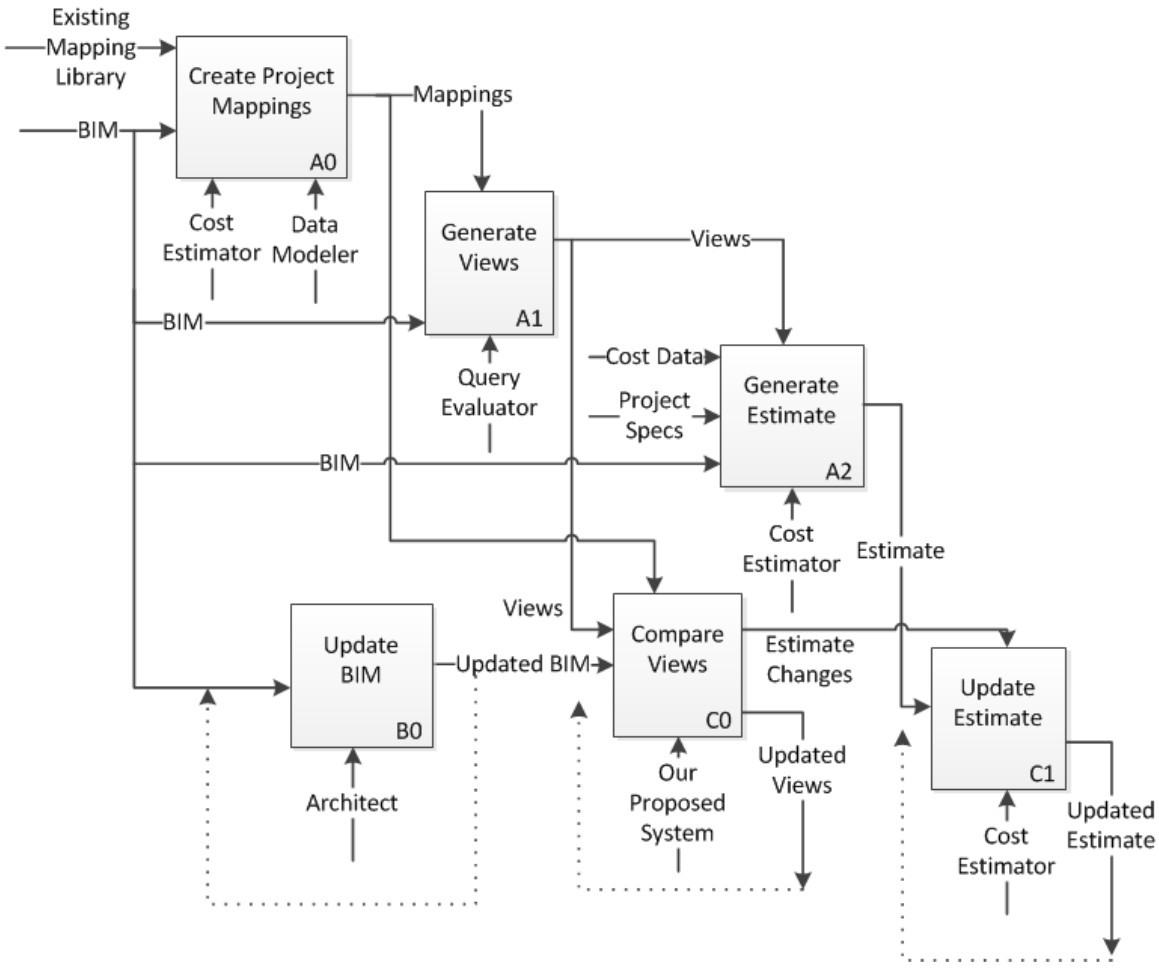


**Fig. 7.** IDEF0 diagram illustrating our approach for updating cost estimates using mappings

*5.1. A0 – Create project mappings*

The first task, for each new building project is to create project mappings (AO). This is a job which requires both a cost estimating expert, and an expert who is familiar with the BIM structure and able to formulate queries expressing the cost estimator's understanding of relevant design features. We call this person a "data modeler". The output is a set of mappings, each associates a query to one or more cost items. The following example illustrates how this task is completed.

*Example 1: Following the example BIM, the cost estimator decides that theatre area is relevant to the theatre millwork cost item. In conjunction with the data modeler, they start to formulate a mapping by creating the query:*

Q1:
//SPACE[LONGNAME='Theatre']/PROPERTYSET/PROPERTY[NAME='Area']/VALUE

*The cost estimator uses the CSI MasterFormat classification system, and so associates this query to the item for theatre millwork with the mapping:*

M1: Q1 ~ (SELECT * FROM ESTIMATE WHERE CSI=6220 AND CODE=0010)

*The result is that query Q1 is related to the item with CSI=6220 and CODE=0010, but since it is a proxy, the specifics of this relationship are not given.*

*Next, the estimator determines that the surface area of elevated slabs is related to several items, so the data modeler creates another query:*

```
Q2:
SUM(
 FOR $X IN //SLAB
 WHERE $X/PROPERTYSET/PROPERTY/VALUE = ‘Suspended Slab’
 RETURN DATA ($X/PROPERTYSET/PROPERTY[NAME='Area']/VALUE))
```

*The cost estimator wants to associate this query with two cost items: the formwork for suspended slabs, whose quantity should be equal to the value returned by Q2, and the item for concrete cleaning, which is related via proxy. The data modeler creates two more mappings:*

M2: Q2 = (SELECT QTY FROM ESTIMATE WHERE CSI=3160 AND CODE=0001)
M3: Q2 ~ (SELECT * FROM ESTIMATE WHERE CSI=3710 AND CODE=0020)

*M3 is similar to M1 in that it expresses a non-specific relationship, but M2 is a quantity takeoff mapping, and expresses that the QTY attribute of the item with CSI=3160 and CODE=0001 should be equal to the value produced by Q2.*

*Lastly, the estimator deems that the number of off-grid concrete columns is relevant to the cost of column formwork. The data modeler creates a query, using the on-grid predicate defined in* [7] *which counts the number of off-grid columns:*

```
Q3:
SUM(
 FOR $X IN //COLUMN
 WHERE on-grid($X) = Boolean(0)
 RETURN 1)
```

*This query is associated with the item for column formwork in the following mapping:*

M4: Q3 ~ (SELECT * FROM ESTIMATE WHERE CSI=3145 AND CODE=11)

Creating mappings is likely to be the most challenging task in implementing our approach. Although we largely leave this problem as future work, we indicate a few promising directions. The database community has long studied the problem of creating schema mappings (see [25] for

a survey). Many of these are likely to be of limited use due to semantic mismatch: the building design contains concepts corresponding to physical building components such as walls, windows and slabs, while the cost estimate is a flat table of items which correspond to work items and materials. One recent area of database research which may be useful is in schema summarization [26]; the ability to summarize large and complex schemas such as IFC would undoubtedly make it easier for the cost estimator and data modeler to create mappings.

Another potential means for simplifying this task is to re-use mappings from previous building projects. That is why we have labeled "Existing Mapping Library" as an input to task A0. In some cases, for example if the BIM is from the same design team as a previous project, a large number of mappings may be able to be re-used. If the BIM is from a different design team but in the same format (e.g. ifcXML), then some of the mappings from previous projects may be reusable, and some may need modifications. Following our example, mapping M1 may not be reusable for future buildings which do not have a theatre. Mappings M2 and M3 may be reusable, but Q2 may need modification, for example if the design team uses the value 'Susp. Slab' instead of 'Suspended Slab' to designate suspended slabs.

With respect to the "pay as you go" philosophy discussed at the end of Section 3, we note that the set of mappings created in task A0 need not be complete, i.e. the cost estimator does not need to encode *all possible* relationships as mappings. Our approach works with existing practices and can be used with as little as one mapping; adding additional mappings yields greater benefits from the system, hence, pay as you go.

### *5.2. A1 – Generate views*

The views are a set of values for each mapping, and are automatically generated from the BIM using query evaluation software given the mapping definitions. Following our example, Q1, Q2 and Q3 would be automatically evaluated by a query evaluation software program (the Query Evaluator), generating four views (one for each mapping) as follows:

V1: 7442.34
V2: 97033.00
V3: 97033.00
V4: 15

In our example, each view consists of only a single value, although in other instances a view may contain a table of values, such as a list of door types and a count of the number of doors of each type. These views are stored along with the mappings on the estimator's computer.

### *5.3. A2 – Generate estimate*

For this task, the cost estimator generates an estimate for the project using whatever methods they choose. The input cost data is organized according to the MasterFormat standard and

contains attributes for the cost code, description, quantity, unit of measure, rate per unit, and total cost. The set of views generated in the previous step A1 act as a control for the estimator to use to generate the cost estimate.

*5.4. B0 – Update BIM*

As the design is evolving, the BIM also gets refined and updated by the party, say the Architect, who has the control over the BIM model. The updated BIM now becomes the reference for comparing views.

*5.5. C0 – Compare views*

The tasks C0 and C1 are repeated each time the building design is changed, and a new estimate is required. It should be noted that the method for triggering this process is inconsequential. The design team may give the cost estimator a new BIM via email, or there might be a shared BIM repository and a new estimate has been requested. In this task, a new set of views is generated by reevaluating the queries in the mappings on the new BIM. Following the above example, an updated set of views may result as follows:

V1': 8250.00
V2': 96452.00
V3': 96542.00
V4': 15

These are then automatically compared with the existing views stored on the cost estimator's computer. For any views that differ, an estimate change is produced depending on the mapping. Following the same example, V1 has increased from 7442.34 to 8250.00. Since M1 relates this value (in a non-specific way) to the item with CSI=6220 and CODE=0010, a change is created which contains the old value, the new value, and the item. V2 and V3 have also changed. In the case of M3, which is a non-specific relationship like M1, a change is created with the old and new values and the item specified in M3. In the case of V2, M2 says that this value should be equal to the QTY attribute of some cost item. The change which is produced specifies that the QTY of this item should be updated to 96452.00. The cost estimator may find it useful to display both of the queries involved in a mapping as well as the corresponding views, shown for mapping M2 in Table 1. This is the default view implemented in our experimental prototype, described in Section 6.

The exact details of this process are described in [16], where the general problem of views which contain tables of values and where the corresponding updates to the right hand side of the mappings (in this case, the cost estimate) are ambiguous are addressed. Once the set of estimate changes are produced, they are passed to the cost estimator for task C1, Update Estimate.

**Table 1.** A side by side layout of the two queries used in mapping M2.

| | |
|---|---|
| SUM(<br>FOR $X IN //SLAB<br>WHERE $X/PROPERTYSET/PROPERTY/VALUE = 'Suspended Slab'<br>RETURN DATA ($X/PROPERTYSET/PROPERTY[NAME='Area']/VALUE)) | SELECT<br>QTY<br>FROM<br>ESTIMATE<br>WHERE<br>CSI=3160 AND<br>CODE=0001 |
| 96452.00 | 97033.00 |

*5.6. C1 – Update estimate*

The estimators automatically receive a set of estimate changes as the output from C0. They are then able to examine each change: the old value, the new value, the cost item in question, and the query associated with the mapping (e.g. as shown in Table 1, where the fact that the value on the left hand side differs from that on the right indicates that the estimate must be updated), and from there decide on how the estimate should be updated. In some cases, such as V2/M2, the estimate can be updated automatically without the estimator's intervention. From informal conversations with practitioners, we learned that a common concern with automation in BIM-based solutions is that the system will take an incorrect or unexpected action without the practitioner's awareness. For this reason our system still informs the estimator (even in the case of automatic updates) what is being changed, and why. These changes could then be traced back to the building design by showing exactly which slabs changed and by how much. This feature provides critical information for the estimator to verify that the suggested changes are correct, or to make additional refinements as necessary.

The updated views generated by task C0 are then stored, replacing the old views generated in task A1. These views are used as input in the next instance the BIM is updated. Similarly, the updated estimate replaces the old estimate and is used for input the next time task C1 is performed.

## 6. Evaluation

We conducted a *user study* with 14 computer science students to evaluate the efficiency and effectiveness of our semi-automatic mapping approach. We selected computer science students because the current stage of prototype development required some database knowledge. We will conduct additional tests in the future with construction practitioners after the next phase of prototype development. In the user study, we wanted to know whether semi-automatic data mapping as implemented in our approach provides a significant time savings over the manual mapping of BIM data. We were also interested to gain additional insights on the types of mappings and updates that our approach is most useful for and understand the users' preference in executing them. An experimental prototype was created for this purpose. The prototype

connects to two databases, *B* (Building Design) and *C* (Cost Estimate), and is configured with a set of mappings in the form of pairs of SQL queries. It has three main screens, B Tables, Mapped Data, and C Tables; only one of which may be showing at any given moment.

'B Tables' show the current instances of building data. The tables are organized into tabs so that one table is visible at a time. 'Mapped Data' are organized into tabs, with one mapping viewable at a time. Each mapping is shown side-by-side, with $\boldsymbol{q_B}$ (the SQL statement) and $\boldsymbol{Q_B}$ (the resulting data) on the left, similarly for *C* on the right. An example is shown in Fig. 8.

'C Tables' show the current instance of cost estimate data, except with the ability to display more than one table at a time in a side-by-side manner. Below these tables, we display information on the translated insertions. An example is shown in Fig. 9. In this case, two tables of *C* are shown side-by-side. An area below is used to display the update translations.

Each time *B* changes, the B Tables screen is refreshed to show the new contents of *B*, and the Mapped Data screen is refreshed to show then new contents of QB for each mapping.

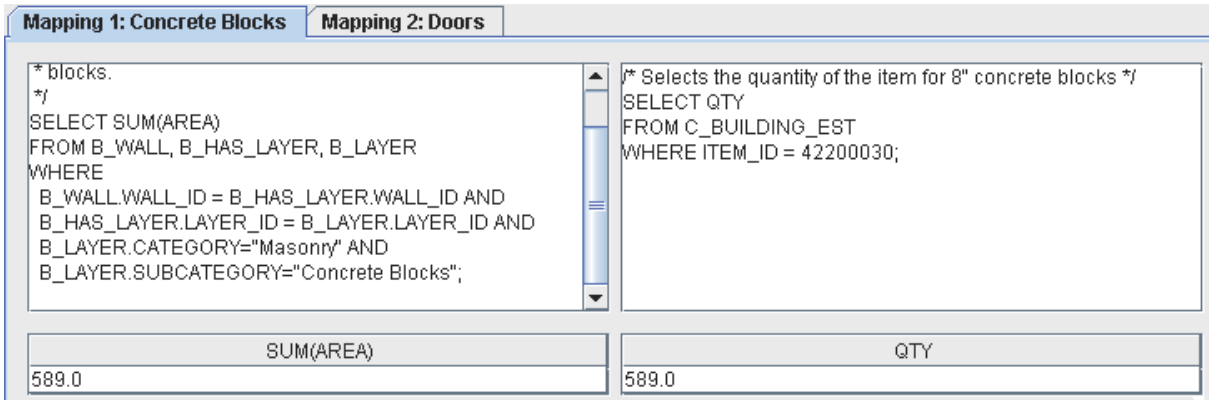


**Fig. 8.** A screen shot of the 'Mapped Data' screen of the data coordination prototype.

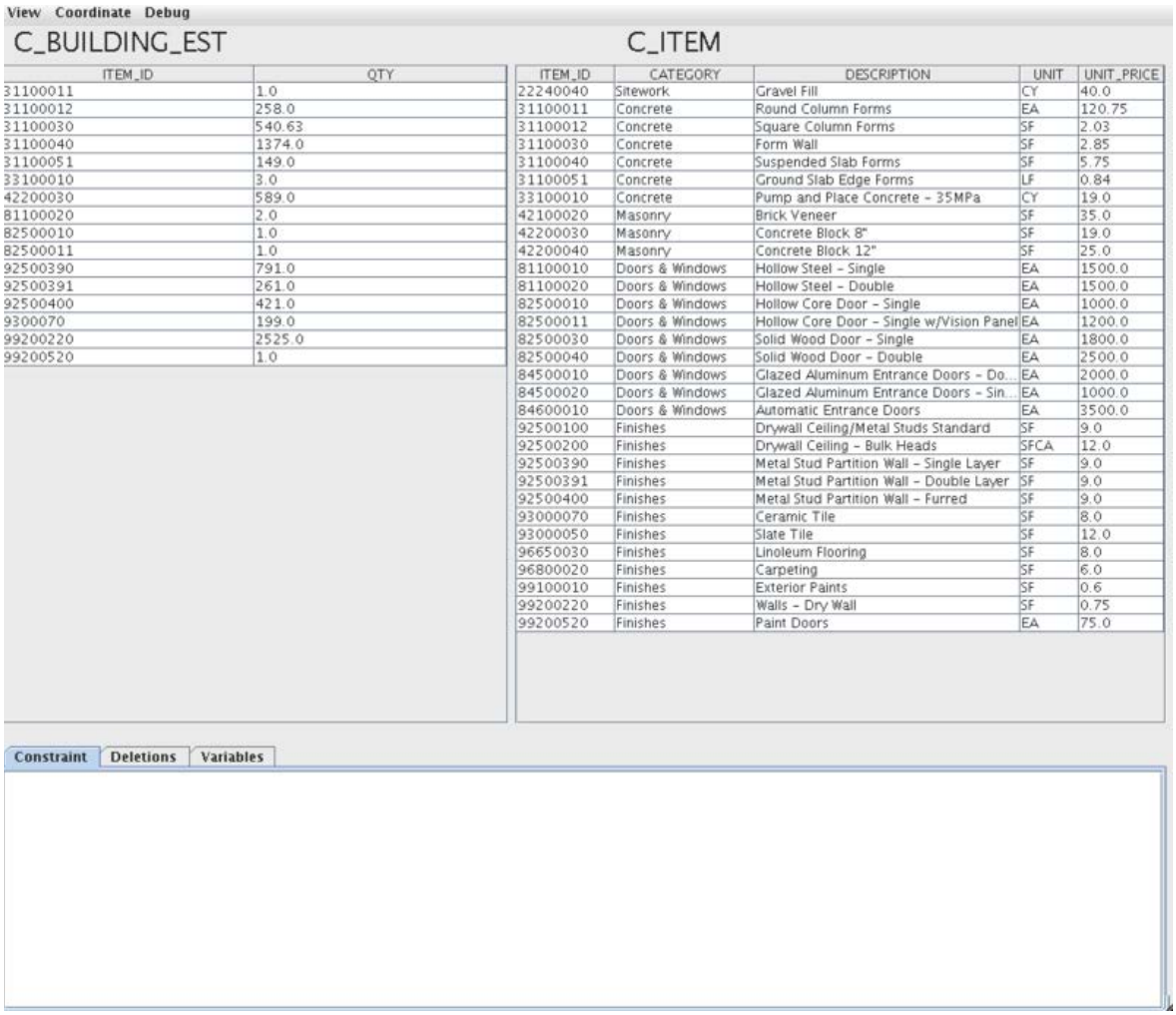


**Fig. 9.** A screen shot of the 'C Tables' screen of the data coordination prototype.

We created a limited scope mapping scenario based on the real world data we obtained for the UBCO construction project described in Section 2. Our intention in choosing this scenario was to be indicative of realistic complexity, but simplified enough to be comprehensible in a single 90-minute experimental session.

Our building database, *B*, was created from a subset of the architectural model of the UBCO building. We exported this model into ifcXML, and created a relational schema corresponding to doors, walls, and wall layers which was then populated by extracting from this ifcXML. The schema for B is shown in Fig. 10. Our cost estimate database, *C*, was created by choosing items from the UBCO estimate which are relevant to the data extracted for *B*. We also created a normal form for this estimate, by using two relations to separate the project-specific quantities of each cost item from the item details. The cost estimate schema is shown in Fig. 11, where C_ITEM contained item details and C_BUILDING_EST contained item quantities for the building in *C*.

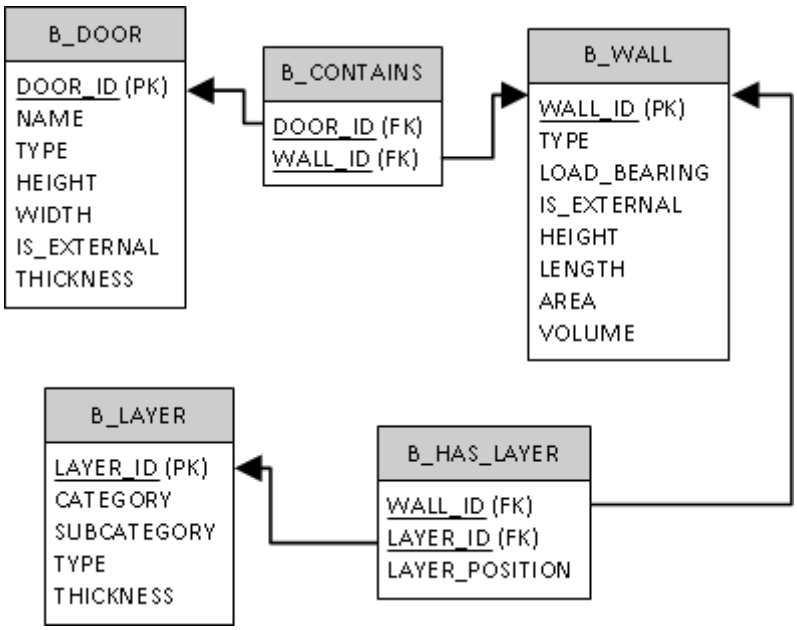


**Fig. 10.** The building design *B* schema.

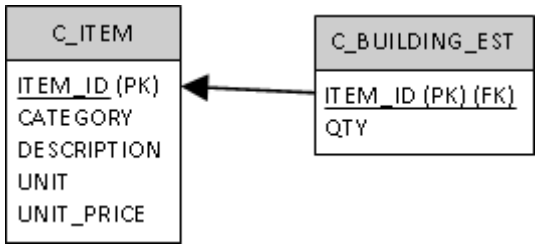


**Fig. 11.** The cost estimate *C* schema.

We chose two relationships representative of real cost estimating scenarios for mappings. Mapping 1 expresses the relationship between an aggregation of the surface area of walls having a layer of "Concrete Blocks" and the quantity of a corresponding item in the cost estimate. We refer these queries as $\boldsymbol{q_{B,1}}$ and $\boldsymbol{q_{C,1}}$. Mapping 2 expresses the fact that the names and counts of each type of door should be the same as the descriptions and quantities of items in the "Doors & Windows" category. The corresponding queries are referred as $\boldsymbol{q_{B,2}}$ and $\boldsymbol{q_{C,2}}$. The queries and data we used for this experiment are provided in detail in [17]

We chose these two mappings to satisfy conflicting design goals: ease of understanding, and representative of realistic complexity. By using a concrete scenario we help make the mappings easy to understand and relate to. The two queries $\boldsymbol{q_{C,1}}$ and $\boldsymbol{q_{C,2}}$ present varying levels of difficulty for our subjects. Updates propagated through query $\boldsymbol{q_{C,1}}$ will adjust the quantity of a particular item.

Before the experiment, each subject read a preparatory script which gave a high level introduction to the scenario, specified their role as the cost estimator, as well as provided a description of each data source and an explanation of the mappings. We recruited 14 subjects (university students) at the University of British Columbia. All of them had a basic understanding of relational database concepts and SQL.

Each subject was given eight distinct update cases. In each update case, a set of updates was applied to *B*, and the subject was required to select the corresponding update to *C*. Subjects were instructed that their goal was to ensure the mappings are satisfied, and should assume that the mappings are correct and complete (i.e. they need not worry about effects on the cost estimate beyond the scope of the mappings).

The first two update cases were warm up tasks, where the preparatory script explained how to use the prototype to choose the correct translation. The results of these tasks were excluded from our results. The remaining six cases are summarized as follows:

1. The area of a single wall is decreased, causing Mapping 1 to be violated. The correct translation is to update the quantity of an item in C_BUILDING_EST.
2. A new door is added, causing Mapping 2 to be violated. A door with the corresponding description exists in C_ITEM, so the correct translation is to insert a single tuple into C_BUILDING_EST with the corresponding quantity.
3. The material of a wall is changed, causing Mapping 1 to be violated. The correct translation is similar to the first case.
4. The type of two doors is changed, causing Mapping 2 to be violated. As opposed to case 2, new tuples must be inserted into both C_ITEM and C_BUILDING_ EST.
5. The width and height of two doors are changed. In this case, mapping is not changed, and so there should be no effect on the estimate.
6. A total of four different changes to *B* cause both mappings to be violated.

In our user study, we also wanted to evaluate whether a combination of assistance strategies provides a significant time savings to raw semi-automatic mapping. For the cost estimating scenario, we have implemented the following assistance strategies within our prototype

- **NONE**. This represents the manual data mapping and coordination scenario. When the assistance level is set to NONE, the Mapped Data screen of the prototype is not available, and no update translations are given. The subject is allowed to browse and query both *B* and *C*, but must analyze the data and use their understanding of the mappings to compose SQL update statements on *C*.
- **LOW**. This represents the “raw” semi-automatic coordination case.
- **HIGH**. This employs the use of assistance strategies as described above: the set of deletions is ranked, suggested values are given for the labeled nulls, and the changes to *B* and each mapping are explicitly shown.

We used a within-subjects protocol (where each subject was presented varying Assistance Levels throughout the experiments) to eliminate the possible skew of a particular subject being matched to a particular assistance level. We measured the amount of time for each subject to complete each update case under different Assistance Level scenario. The update cases 1 and 3, as well as 2 and 4 were grouped due to their similarity. The results are shown in Fig. 12. The results indicate that the completion time for each update tends to decrease as the level of assistance provided for data coordination and mapping between design data and cost estimate is increased. An independent sample t-test confirmed that the time saving due to the assistance level provided were significant, thereby confirming the usability of our mapping approach developed in this research.

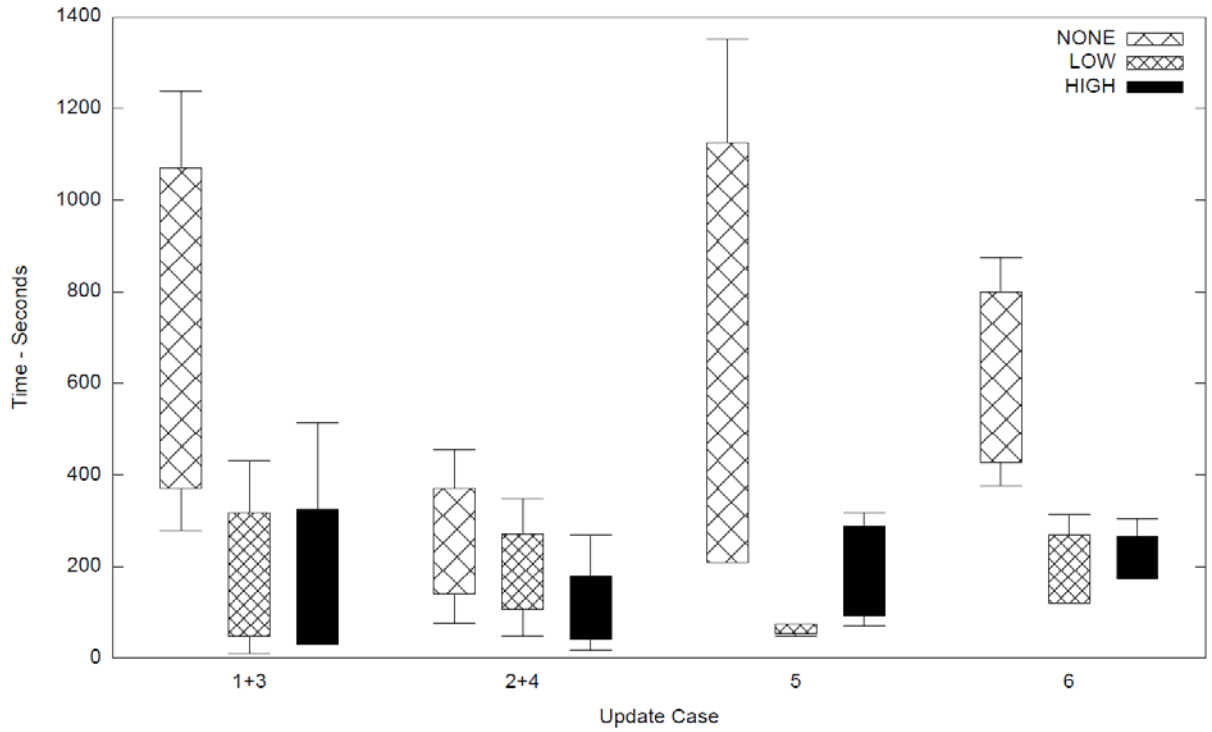


**Fig. 12.** The time taken for subjects to complete each update case, based on assistance level. Each bar is centered on the mean, and has height of twice the standard deviation. The whiskers indicate the minimum and maximum values.

For Update Cases 1 and 3, which cause a violation of Mapping 1, there was a large difference between the manual and semi-automatic approaches. In contrast, there is a much narrower difference between the manual and semi-automatic approaches for Update Cases 2 and 4, which violate mapping 2 and result in ambiguous update translations. Update Case 5 is the instance where *B* changes but both mappings are satisfied. All subjects who had an Assistance level of LOW for this case were able to complete it fairly quickly. Interestingly, subjects who had an Assistance level of HIGH took longer, possibly because they were distracted by explicitly seeing the changes to *B*. Subjects coordinating manually for this case were varied. We saw the clearer

evidence of the benefit of semi-automatic approach over manual for the Update Case 6, which violates both mappings. However, due to the burden of reviewing the changes to *B*, the HIGH assistance level subjects did not outperform the LOW subjects.

Our observation also highlighted the importance of understanding the data. From our interviews with practitioners, we learned that a common concern with computer automation in BIM-based solutions is that the system will take an incorrect or unexpected action without the practitioner's awareness. It is very important for the cost estimator to understand why a given update has been performed or is suggested. This allows the cost estimator to verify that the suggested changes are correct, and if not, help the cost estimator refine the mappings if necessary. Almost all subjects in our user study agreed that being able to query *B* (or the design data specifically, with the mappings) is extremely useful when doing manual coordination. Surprisingly, all except for one subject found it very helpful to see the changes to *B* highlighted with the HIGH assistance level. Since these changes are irrelevant to the task (other than how the mapping is affected), we suspect that knowing the *B* changes appeared to be useful because it provides a slightly deeper insight into what has happened, rather than providing any actionable data. Understanding this increases our ability to help building designers be more flexible; by providing a method by which the cost estimate can be updated automatically, more changes can be considered.

## 7. Conclusion

Updating a building's cost estimate corresponding to multiple changes in the design is a major challenge for estimators. These challenges are exacerbated when the revised designs are issued without any indication of what has changed, resulting in a lot of manual work looking for changes, and determining the impacts on the cost estimate. The heterogeneity between design and cost data, coupled with the lack of software interoperability and standardization of cost data and cost estimating practices has led to the slow adoption of BIM in these important business practices.

In this research, we proposed and implemented an approach to coordinate design and cost information that uses declarative mappings to express exact relationships between BIM objects and cost data. The approach uses queries on the building design to populate views on the building design and cost estimate in order to generate the set of possible updates which satisfy a mapping. The user can select an updated view from the set of possible updates through assistance strategies provided. We designed and validated our approach through extensive studies of two real-world construction projects and through interviews with domain experts. The user study conducted with university students demonstrated that our system can increase work efficiency and effectiveness in maintaining a cost estimate as the design changes. Additional tests are needed to demonstrate the usability of our approach with professional cost estimators.

We believe that our approach can be useful to: (1) decrease the time needed for an estimator to update their estimate given an updated design; (2) generate a more accurate estimate by taking into account the implicit design conditions that may otherwise go unnoticed; and (3) provide expanded coverage of the cost estimate by allowing a broader class of spatial and geometric relationships of design objects to be captured. All of these in turn can enable further exploration of more design alternatives, leading to a more flexible solution. Future work is needed to develop more advanced assistance strategies and to develop heuristics to help the end user choose the right set of updates. More work is also needed to assist the user in creating mappings, to support the reuse of mappings and in providing a good user interface to shield users from needing to create complex queries. Indeed, one approach that we would like to explore is how to allow the user to automatically graphically create very simple mappings, which would likely be sufficient for many cases and then have support to create very complex mappings (as in [27]) in order to solve the remaining cases. Finally, additional tests are also needed on other projects and for different estimators to broaden the range of queries that can be supported.

**References**


[1] W. Kraus, S. Watt, P. Larson, Challenges in Estimating Costs Using Building Information Modeling, AACE International Transactions (2007) IT11-IT13.

[2] C.M. Eastman, P. Teicholz, R. Sacks, K. Liston, BIM handbook : a guide to building information modeling for owners, managers, designers, engineers, and contractors, Wiley, Hoboken, N.J., 2008.

[3] D.L. Spooner, M. Hardwick, Using views for product data exchange, Computer Graphics and Applications, IEEE 17 (5) (1997) 58-65.

[4] U. Isikdag, J. Underwood, Two design patterns for facilitating Building Information Model-based synchronous collaboration, Automation in Construction 19 (5) (2010) 544-553.

[5] CostX Construction Estimating Software, Exactal Technologies, http://www.exactal.com/products/costX/, 2013.

[6] Innovaya Inc., Innovaya, http://www.innovaya.com/, 2012.

[7] M.P. Nepal, S. Staub-French, R. Pottinger, A. Webster, Querying a building information model for construction-specific spatial information, Advanced Engineering Informatics 26 (4) (2012) 904-923.

[8] T.L. McCuen, C.L. Del Puerto, Cost savings achieved through changing processes for cost estimating in building information modeling, 55th Annual Meeting of the Association for

the Advancement of Cost Engineering, Vol. 1, Association for the Advancement of Cost Engineering, Anaheim, CA, United states, 2011, p. 10.

[9] Z. Shen, R.R. Issa, Quantitative evaluation of the BIM-assisted construction detailed cost estimates, Journal of Information Technology in Construction (ITcon) 15 (2010) 234-257.

[10] T.L. McCuen, Underdeveloped and Underutilized: Cost Estimating in BIM, 54th Annual Meeting of the American Association of Cost Engineers International 2010, Vol. 1, Association for the Advancement of Cost Engineering, Atlanta, GA, 2010, p. 9.

[11] J.J. Hannon, Estimators' Functional Role Change with BIM, AACE International Transactions (2007) IT31-IT38.

[12] T.L. McCuen, The quantification process and standards for BIM, 53rd AACE International Annual Meeting 2009, Association for the Advancement of Cost Engineering, Seattle, WA, 2009, p. 11.

[13] C. Eastman, Y. Jeong, R. Sacks, I. Kaner, Exchange Model and Exchange Object Concepts for Implementation of National BIM Standards, Journal of Computing in Civil Engineering 24 (1) (2010) 25-34.

[14] M. Venugopal, C.M. Eastman, R. Sacks, J. Teizer, Semantics of model views for information exchanges using the industry foundation class schema, Advanced Engineering Informatics 26 (2) (2012) 411-428.

[15] T. Hartmann, H. van Meerveld, N. Vossebeld, A. Adriaanse, Aligning building information model tools and construction management methods, Automation in Construction 22 (2012) 605-613.

[16] M. Lawrence, R. Pottinger, S. Staub-French, Data Coordination: Supporting Contingent Updates, Proceedings of the VLDB Endowment 4 (11) (2011).

[17] M. Lawrence, Data Coordination, PhD thesis, Department of Computer Science, The University of British Columbia, 2012.

[18] P.A. Bernstein, A.Y Halevy, R.A Pottinger, Model management: managing complex information structures. SIGMOD Record, 29 (4) (2000) 55-63.

[19] A. Doan, A.Y. Halevy, Semantic-Integration Research in the Database Community: A Brief Survey, AI Magazine 26 (1) (2005) 83-94.

[20] A. Halevy, M. Franklin, D. Maier, Principles of dataspace systems, Proceedings of the twenty-fifth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, ACM, Chicago, IL, 2006, pp. 1-9.

[21] J. Horak, L. Jurikovska, M. Umlauf, J. Jezek, M. Hanzlova, S. Zlatanova, HUMBOLDT Alignment Editor, Symposium GIS Ostrava 2011, Ostrava, Czech Republic, 2011.

[22] A. Gupta, I.S. Mumick, Materialized views: techniques, implementations, and applications, MIT press, 1999.

[23] M. Friedman, A. Levy, T. Millstein, Navigational plans for data integration, Proceedings of the National Conference on Artificial Intelligence, John Wiley & Sons Ltd., 1999, pp. 67-73.

[24] A. Borrmann, E. Rank, Specification and implementation of directional operators in a 3D spatial query language for building information models, Advanced Engineering Informatics 23 (1) (2009) 32-44.

[25] E. Rahm, P.A. Bernstein, A survey of approaches to automatic schema matching, The VLDB Journal 10 (4) (2001) 334-350.

[26] C. Yu, H. Jagadish, Schema summarization, Proceedings of the 32nd international conference on very large data bases, VLDB Endowment, 2006, pp. 319-330.

[27] R. Miller, M. Hernandez, L. Hass, L. Yan, C. Ho, R. Fagin, L. Popa, The Clio project: managing heterogeneity, SIGMOD Record, 30 (1) (2001) 78-83.